UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/535,065	10/24/2005	Catherine Robert	S1022.81242US00	1848

46329 7590 07/25/2008

STMicroelectronics Inc.
c/o WOLF, GREENFIELD & SACKS, P.C.
600 Atlantic Avenue
BOSTON, MA 02210-2206

EXAMINER	
VICARY, KEITH E	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
07/25/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/535,065	ROBERT ET AL.	
	Examiner	Art Unit	
	Keith Vicary	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *22 May 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-11* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-11* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-11 are pending in this office action and presented for examination. Claims 1, 3, 7-9, and 11 have been amended by amendment filed 5/22/2008.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claim 9 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

4. Claim 9 recites the limitation "assigning a first value to a first set of bits of the explicit jump message" in lines 5-6. However, given that it is the assigning of a first value to a first set of bits that characterizes a digital message as an explicit jump message, it is indefinite as to how the assigning can be done to an already existing explicit jump message. Examiner recommends the language be modified in a manner akin to the limitations regarding the implicit jump message (e.g. using "to provide").

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and

> the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-11 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nexus 5001 Forum: Standard for a Global Embedded Processor Debug Interface (Nexus 5001 Forum) in view of Argade et al. (Argade) (US 5724505).

7. Nexus 5001 Forum is cited by the applicant in IDS paper filed 5/13/2005.

8. Argade is cited by the applicant in IDS paper filed 5/13/2005.

9. **Consider claim 1**, Nexus 5001 Forum discloses a method for transmitting digital messages (page 52 of 150, section 6.2, transmission), on execution of an instruction sequence by a microprocessor (page 59 of 150, last paragraph, message is output by the target processor whenever there is a change of program flow), through output terminals of a monitoring circuit integrated on the microprocessor (Table 7-1, Auxiliary Pins required per interface), at least one of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages), the method comprising, the steps of: determining whether the jump is associated with a jump instruction of the instruction sequence for which data representative of the destination instruction address of the jump is explicitly indicated in the instruction (Table 6-6 and Table 6-7, which shows the resulting messages based on the determination; a determination is inherent for the appropriate message to be sent); if yes, assigning a first value to a first set of bits of at least one digital message to provide

an explicit jump message, and if not, assigning a second value to the first set of bits to provide an implicit jump message(Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3; when no, Table 6-7 shows that the TCODE set of bits will be equal to 4); transmitting the at least one digital message (page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose when the first set of bits is at the second value, *providing an additional field in the implicit jump message, the additional field comprising a second set of bits, and assigning to the second set of bits a third value identifying the jump as an implicit jump from among several types of implicit jumps*.

On the other hand, Argade does disclose of identifying a jump as an implicit jump from among several types of implicit jumps by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2, and type_3 in col. 5, lines 49-67).

Argade's teaching of a field which identifies the jump from among several types of jumps enables the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above.

Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the environment of Agrade is analogous to the environment of Nexus 5001 Forum, as Agrade also discloses of a method for transmitting digital messages (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal), on execution of an instruction sequence by a microprocessor (col. 4, lines 30-31, 39; the program trace), through output terminals (col. 4, 59-65, JTAG interface and port) of a monitoring circuit (col. 4, lines 51, 59-65, HDS block) integrated on the microprocessor (col. 4, line 39; Figure 1 also shows the JTAG interface (24), JTAG port (44), and monitoring circuit (26) clearly inside the microprocessor (10)), at least one of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (col. 5, lines 39-45, wherein a discontinuity corresponds to the jump, and the INSTR_TYPE and its corresponding address is the characteristic data, which is part of the digital message as seen again in col. 6, lines 24-27), comprising, for the transmission of the at least one digital message, the steps of: determining whether the jump is associated with a jump instruction of the instruction sequence for which data representative of a destination instruction address of the jump is explicitly indicated in the instruction (col. 5, lines 65-67, the third discontinuity type includes an absolute

address instruction jump or call); providing a field comprising a second set of bits of the at least one digital message and assigning to the second set of bits a third value identifying the jump from among several types of jumps (col. 5, lines 39-45, the INSTR_TYPE, which is part of the digital message as above in col. 6, lines 24-27); and transmitting the digital message (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal).

Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the teaching of a jump type field of Agrade could be additionally added to the program trace, indirect branch message of Nexus 5001 Forum, Table 6-7 without also adding it to the program trace, direct branch message of Nexus 5001 Forum, Table 6-6. Agrade discloses that his type_3 discontinuity does not need to record an address (Agrade, col. 6, lines 1-4), which fits into Nexus 5001 Forum's teaching of his program trace, direct branch message of Table 6-6, which does not have an address field. Furthermore, Nexus 5001 Forum, Tables 6-6 and 6-7, shows that the two different types of messages are already of different length; it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the jump type field would be added to the indirect branch message without affecting the direct branch message.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus

5001 Forum in order to enable the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

10. **Consider claim 7**, Nexus 5001 Forum discloses means for detection of a jump on execution of an instruction sequence by the microprocessor (Nexus 51001 Forum, Table 6-6 and 6-7); means for storing data characteristic of the detected jump (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages); means for generating a digital message based on the stored characteristic data, the digital message comprising a first set of bits set to a first value if the jump is associated with a jump instruction of the instruction sequence for which data representative of a destination instruction address of the jump are explicitly indicated in the instruction, wherein the digital message is an explicit jump message, and the first set of bits set to a second value in the opposite case, wherein the digital message is an implicit jump message (Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3; when no, Table 6-7 shows that the TCODE set of bits will be equal to 4); and means for transmitting the generated digital message (page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose when the first set of bits is set to the second value, *providing an additional field in the implicit jump message, the*

additional field comprising a second set of bits, with the second set of bits set to a third value identifying an implicit jump from among several implicit jump types.

On the other hand, Argade does disclose of identifying an implicit jump from among several implicit jump types by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2, and type_3 in col. 5, lines 49-67).

Argade's teaching of a field which identifies the jump from among several types of jumps enables the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the environment of Agrade is analogous to the environment of Nexus 5001 Forum, as Agrade also discloses of a method for transmitting digital messages (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal), on execution of an instruction sequence by a microprocessor (col. 4, lines 30-31, 39; the program trace), through

output terminals (col. 4, 59-65, JTAG interface and port) of a monitoring circuit (col. 4, lines 51, 59-65, HDS block) integrated on the microprocessor (col. 4, line 39; Figure 1 also shows the JTAG interface (24), JTAG port (44), and monitoring circuit (26) clearly inside the microprocessor (10)), at least one of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (col. 5, lines 39-45, wherein a discontinuity corresponds to the jump, and the INSTR_TYPE and its corresponding address is the characteristic data, which is part of the digital message as seen again in col. 6, lines 24-27), comprising, for the transmission of the at least one digital message, the steps of: determining whether the jump is associated with a jump instruction of the instruction sequence for which data representative of a destination instruction address of the jump is explicitly indicated in the instruction (col. 5, lines 65-67, the third discontinuity type includes an absolute address instruction jump or call); providing a field comprising a second set of bits of the at least one digital message and assigning to the second set of bits a third value identifying the jump from among several types of jumps (col. 5, lines 39-45, the INSTR_TYPE, which is part of the digital message as above in col. 6, lines 24-27); and transmitting the digital message (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal).

Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the teaching of a jump type field of Agrade could be additionally added to the program trace, indirect branch message of Nexus 5001 Forum, Table 6-7 without also adding it to the program trace, direct branch message of Nexus 5001 Forum, Table 6-6. Agrade discloses that his type_3 discontinuity does not need to record an address (Agrade, col. 6, lines 1-4), which fits into Nexus 5001 Forum's teaching of his program trace, direct branch message of Table 6-6, which does not have an address field. Furthermore, Nexus 5001 Forum, Tables 6-6 and 6-7, shows that the two different types of messages are already of different length; it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the jump type field would be added to the indirect branch message without affecting the direct branch message.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to enable the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

11. **Consider claim 8**, Nexus 5001 Forum discloses detecting a jump in the execution of the instruction sequence from an initial instruction to a jump destination instruction, wherein the jump destination instruction is different from an instruction

following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages, the detection is inherent in the transmission of these messages); generating at least one digital message upon the detection of the jump (Table 6-6 and 6-7), if the jump is implicit, generating the at least one digital message as an implicit jump message (Table 6-7 shows that the TCODE set of bits will be equal to 4) if the jump is not implicit, generating the at least one digital message as an explicit jump message (Table 6-6); and transmitting the at least one digital message (page 52 of 150, section 6.2, transmission).

However, Nexus 5001 Forum does not explicitly disclose if the jump is implicit, *providing an additional field in the implicit jump, wherein the additional field includes a value identifying a type of the implicit jump*.

On the other hand, Argade does disclose of identifying a jump from among several types of implicit jumps by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2, and type_3 in col. 5, lines 49-67).

Argade's teaching of a field which identifies the jump from among several types of jumps enables the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001

Forum in order to successfully debug most programs, as explained above. Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the environment of Agrade is analogous to the environment of Nexus 5001 Forum, as Agrade also discloses of a method for transmitting digital messages (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal), on execution of an instruction sequence by a microprocessor (col. 4, lines 30-31, 39; the program trace), through output terminals (col. 4, 59-65, JTAG interface and port) of a monitoring circuit (col. 4, lines 51, 59-65, HDS block) integrated on the microprocessor (col. 4, line 39; Figure 1 also shows the JTAG interface (24), JTAG port (44), and monitoring circuit (26) clearly inside the microprocessor (10)), at least one of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (col. 5, lines 39-45, wherein a discontinuity corresponds to the jump, and the INSTR_TYPE and its corresponding address is the characteristic data, which is part of the digital message as seen again in col. 6, lines 24-27), comprising, for the transmission of the at least one digital message, the steps of: determining whether the jump is associated with a jump instruction of the instruction sequence for which data representative of a destination instruction address of the jump is explicitly indicated in

the instruction (col. 5, lines 65-67, the third discontinuity type includes an absolute address instruction jump or call); providing a field comprising a second set of bits of the at least one digital message and assigning to the second set of bits a third value identifying the jump from among several types of jumps (col. 5, lines 39-45, the INSTR_TYPE, which is part of the digital message as above in col. 6, lines 24-27); and transmitting the digital message (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal).

Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the teaching of a jump type field of Agrade could be additionally added to the program trace, indirect branch message of Nexus 5001 Forum, Table 6-7 without also adding it to the program trace, direct branch message of Nexus 5001 Forum, Table 6-6. Agrade discloses that his type_3 discontinuity does not need to record an address (Agrade, col. 6, lines 1-4), which fits into Nexus 5001 Forum's teaching of his program trace, direct branch message of Table 6-6, which does not have an address field. Furthermore, Nexus 5001 Forum, Tables 6-6 and 6-7, shows that the two different types of messages are already of different length; it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the jump type field would be added to the indirect branch message without affecting the direct branch message.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to enable the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

12. **Consider claim 11**, Nexus 5001 Forum discloses a monitoring circuit integrated on a microprocessor for (Table 7-1, Auxiliary Pins required per interface; also, some form of monitoring circuit is inherent given that trace messages are being sent based on program flow): detecting, on execution of an instruction sequence by the microprocessor, a jump from an initial instruction to a jump destination instruction, wherein the jump destination instruction is different from an instruction following the initial instruction in the instruction sequence (page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages, the detection is inherent in the transmission of these messages); if the jump is not implicit, providing the at least one digital message as an explicit jump message (see above citation); an analysis tool to reconstitute the instruction sequence based on the at least one digital message; and at least one monitoring terminal to provide the at least one digital message from the monitoring circuit to the analysis tool (Table 7-1, Auxiliary Pins required per interface; also, some form of monitoring circuit is inherent given that trace messages are being sent based on

program flow; page 5 of 150, last paragraph, program trace visibility, development tools, page 51 of 150, first paragraph, the tool; page 58 of 150, section 6.4.4, program trace).

However, Nexus 5001 Forum does not explicitly disclose if the jump is implicit, *providing an additional field in at least one digital message to provide the at least one digital message as an implicit jump message transmitted on the execution of the instruction sequence by the microprocessor, wherein the additional field includes a value identifying a type of the implicit jump*.

On the other hand, Argade does disclose of identifying a jump from among several types of implicit jumps by assigning to a set of bits a specific value (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2, and type_3 in col. 5, lines 49-67).

Argade's teaching of a field which identifies the jump from among several types of jumps enables the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to successfully debug most programs, as explained above. Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the environment of Agrade is analogous to the environment of Nexus 5001 Forum, as Agrade also discloses of a method for

transmitting digital messages (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal), on execution of an instruction sequence by a microprocessor (col. 4, lines 30-31, 39; the program trace), through output terminals (col. 4, 59-65, JTAG interface and port) of a monitoring circuit (col. 4, lines 51, 59-65, HDS block) integrated on the microprocessor (col. 4, line 39; Figure 1 also shows the JTAG interface (24), JTAG port (44), and monitoring circuit (26) clearly inside the microprocessor (10)), at least one of said digital messages being representative of characteristic data stored by the monitoring circuit on detection of a jump in the execution of an instruction sequence from an initial instruction to a destination instruction different from an instruction following the initial instruction in the instruction sequence (col. 5, lines 39-45, wherein a discontinuity corresponds to the jump, and the INSTR_TYPE and its corresponding address is the characteristic data, which is part of the digital message as seen again in col. 6, lines 24-27), comprising, for the transmission of the at least one digital message, the steps of: determining whether the jump is associated with a jump instruction of the instruction sequence for which data representative of a destination instruction address of the jump is explicitly indicated in the instruction (col. 5, lines 65-67, the third discontinuity type includes an absolute address instruction jump or call); providing a field comprising a second set of bits of the at least one digital message and assigning to the second set of bits a third value identifying the jump from among several types of jumps (col. 5, lines 39-45, the

INSTR_TYPE, which is part of the digital message as above in col. 6, lines 24-27); and transmitting the digital message (col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal).

Furthermore, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the teaching of a jump type field of Agrade could be additionally added to the program trace, indirect branch message of Nexus 5001 Forum, Table 6-7 without also adding it to the program trace, direct branch message of Nexus 5001 Forum, Table 6-6. Agrade discloses that his type_3 discontinuity does not need to record an address (Agrade, col. 6, lines 1-4), which fits into Nexus 5001 Forum's teaching of his program trace, direct branch message of Table 6-6, which does not have an address field. Furthermore, Nexus 5001 Forum, Tables 6-6 and 6-7, shows that the two different types of messages are already of different length; it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the jump type field would be added to the indirect branch message without affecting the direct branch message.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teaching of Agrade with the invention of the Nexus 5001 Forum in order to enable the capturing of information about whether certain types of instructions were executed, including conditionally executed instructions; a trace of

these instructions may be very important in debugging most programs (Agrade, col. 3, lines 17-21).

13. **Consider claim 2**, Nexus 5001 Forum discloses the step of assigning to a third set of bits of the at least one digital message a value corresponding to a number of instructions executed by the microprocessor since a last executed instruction of the instruction sequence for which a digital message associated with a jump was transmitted (page 59, Table 6-6 and 6-7, the I-CNT field).

14. **Consider claim 3**, Nexus 5001 Forum discloses the step of, if the first set of bits is at the second value, assigning to a fourth set of bits of the implicit jump message a value representative of the address of the destination instruction (Nexus 5001 Forum, Table 6-7, U-ADDR).

15. **Consider claim 4**, Argade discloses a jump type corresponds to a jump resulting from a jump instruction of the instruction sequence containing a reference of a register in which are stored data representative of the destination instruction address (col. 5, lines 59-65, register indirect jump or call).

16. **Consider claim 5**, Argade discloses a jump type corresponds to a jump forced by the microprocessor, the destination instruction corresponding to an instruction comprising a series of specific instructions which are different from instructions of the instruction sequence (col. 5, lines 49-52, hardware interrupt).

17. **Consider claim 6**, Argade discloses a jump type corresponds to a jump forced by the microprocessor, the destination instruction being an instruction of the instruction sequence (col. 5, lines 65-67 and col. 6, line 1; relative…address jump or call).

18. **Consider claim 9**, Nexus 5001 Forum and Argade disclose determining whether the jump is associated with a jump instruction of the instruction sequence explicitly indicating an address of the jump destination instruction (Nexus 5001 Forum, page 59 of 150, Table 6-6 and 6-7, branch message and indirect branch messages; it is inherent a determination takes places as either of the two messages are formed as a result of whether the address is explicitly indicated or not); if it is determined that the jump instruction explicitly indicates the address of the jump destination instruction, assigning a first value to a first set of bits of the explicit jump message (Nexus 5001 Forum, Table 6-6 and Table 6-7, when yes, Table 6-6 shows that the TCODE set of bits will be equal to 3); and if it is determined that the jump instruction does not explicitly indicate the address of the jump destination instruction: assigning a second value to the first set of bits to provide the implicit jump message (Nexus 5001 Forum, when no, Table 6-7 shows that the TCODE set of bits will be equal to 4); and assigning to the additional field comprising a second set of bits a third value identifying the type of the implicit jump (col. 5, lines 39-45, the INSTR_TYPE, which is part of a digital message of lines 24-27, with the types of jumps being type_1, type_2, and type_3 in col. 5, lines 49-67).

19. **Consider claim 10**, Nexus 5001 Forum and Argade disclose the at least one digital message is transmitted through output terminals of a monitoring circuit integrated on the microprocessor (Nexus 5001 Forum, Table 7-1, Auxiliary Pins required per interface; Argade, col. 6, lines 24-27, wherein the signal is the digital message; col. 6, lines 46-48, wherein the shifting to the JTAG interface is the first phase of transmittal; and col. 4, lines 63-65, wherein the movement into the JTAG port and the debug host computer is the second phase of transmittal, col. 4, 59-65, JTAG interface and port, col. 4, lines 51, 59-65, HDS block)

Response to Arguments

20. Applicant's arguments filed 5/22/2008 have been fully considered but they are not persuasive.

21. Applicant argues on page 8 that Argade only teaches of a signal which indicates the type of each instruction executed, and not a digital message. However, it is first noted that a signal can be broadly interpreted to be a type of digital message. More importantly, Nexus teaches of the digital message. Nexus's digital message, as modified by Argade's teaching of implicit jump type information, teaches the overall limitation of a digital message containing implicit jump type information.

Applicant argues that it is not clear why the difference in the length of the direct and indirect branch messages of Nexus would suggest that an additional field needs to be added to an indirect branch message, since this would result in further increase in the difference in length between the two messages. Applicant relatedly argues that it is

not clear why, even if the references were combined, would an additional field be added only to an implicit jump message and not to both implicit and explicit jump messages. However, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that Argade's teaching of implicit jump information is only applicable to implicit jumps and not explicit jumps. Therefore, it would have been readily recognized to one of ordinary skill in the art there would be no reason to add an additional implicit jump information field to an *explicit* jump message. Examiner noted that the direct and indirect branch digital messages were already different lengths in Nexus to highlight that adding an additional implicit jump information field only to the indirect branch digital message is consistent with the environment of Nexus that does not require both the direct and indirect branch digital messages to be of the same length.

Argade teaches of identifying a jump from several types of jumps and the benefits of doing so, as explained in the rejection above. It would have been readily recognized to one of ordinary skill in the art at the time of the invention that this general concept of Argade of conveying jump information to a user or debugger is not dependent on the exact manner in which that jump information is physically conveyed to the user. For example, jump information being conveyed using USB protocols, Firewire protocols, and so forth would be obvious variants of Argade. Similarly, jump information being conveyed by digital messages as disclosed by Nexus would also be an obvious variant, especially given that the digital messages of Nexus are also used for debugging purposes and already identify direct and indirect jump instructions. Moreover, given that

digital messages are necessarily composed of fields of bits, it would have been readily recognized to one of ordinary skill in the art at the time of the invention that the combination of the teaching of Argade into the invention of Nexus, which provides for additional indirect jump information, would also entail additional bits.

Conclusion

22. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

23. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Keith Vicary whose telephone number is (571)270-1314. The examiner can normally be reached on Monday - Thursday, 6:15 a.m. - 5:45 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on 571-272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Eddie P Chan/
Supervisory Patent Examiner, Art Unit 2183

/Keith Vicary/
Examiner, Art Unit 2183